МІНІСТЕРСТВО ОСВІТИ І НАУКИ,
МОЛОДІ ТА СПОРТУ УКРАЇНИ

Національний технічний університет
«Харківський політехнічний інститут»

МЕТОДИЧНІ ВКАЗІВКИ
до лабораторної роботи
«Рядкові дані у Delphi»

з курсу «Програмування»
для студентів напряму 6.040302 – Інформатика
(спеціалізація «Соціальна інформатика»)

Затверджено редакційно-видавничою
радою університету,
протокол № 2 від 01.12.10.

Харків НТУ «ХПІ» 2011

Методичні вказівки до лабораторної роботи «Рядкові дані у Delphi» з курсу «Програмування» для студентів напряму 6.040302 – Інформатика (спеціалізація «Соціальна інформатика») / Уклад. М. І. Безменов. – Х. : НТУ «ХПІ», 2011. – 21 с.

Укладач М. І. Безменов

Рецензент Л. М. Любчик

Кафедра системного аналізу і управління

ВСТУП

Обробка текстової інформації є неможливою без використання рядкових типів. Крім того, особливістю програм, написаних у візуальному середовищі Delphi, є те що введення та виведення даних орієнтоване на використання рядків.

Метою даної лабораторної роботи є освоєння методики обробки рядкових даних у програмах, написаних мовою Delphi.

1. ТЕОРЕТИЧНІ ОСНОВИ

1.1. Рядкові типи у Delphi 2009

У Delphi визначено тип `Char`, значеннями якого є символи (літери, цифри, різні знаки). При цьому можливе використання як однобайтових, так двобайтових символів. Відповідно до цього визначано два символьні типи – `AnsiChar` та `WideChar`. Тип `Char` в залежності від версії є еквівалентним одному з цих типів, а саме, у версіях, нажчих за Delphi 2009, – типу `AnsiChar`, а у Delphi 2009 та Delphi 2010 – типу `WideChar`. Часто ж потрібно працювати не з окремими символами, а з їх рядками. У Delphi 2009 визначено кілька типів для опису змінних, у яких можуть зберігатися рядки символів:

`UnicodeString` або **`string`** – широкі рядки;

`AnsiString` – довгі рядки;

`ShortString` або **`string`**`[N]` – короткі рядки;

`WideString` – широкі рядки.

Крім того, для роботи з рядковими даними використовують типи `PChar`, `PWideChar`, `PAnsiChar` і масиви символів (**`array of`** `Char`, **`array of`** `WideChar` та **`array of`** `AnsiChar`) з нижньою межею індексу, що дорівнює 0 (символьні масив з нульовою базою).

Будь-який рядок може трактуватися як масив символів. При цьому короткі рядки можна розглядати як масив з кількістю елементів, зумовленою довжиною рядка, а інші види рядків – як масиви символів, що обмежуються першою появою символу `#0`.

Оголошення рядкових змінних здійснюється звичайним способом. Наприклад:

```
var
  StrUnicode : UnicodeString;
```

```
Str         : string;
Str         : AnsiString;
StrShort    : ShortString;
Str6        : string[6];
StrWide     : WideString;
StrPCh      : PChar;
StrArr      : array [0..1000] of Char;
```

Кращим для більшості цілей є тип `UnicodeString`.

У виразах рядки різних типів можуть зустрічатися спільно. Перетворення типів при цьому відбувається автоматично. Однак рядки, що підставляються замість **var**- та **out**-параметрів при звертанні до процедур і функцій, повинні мати тип, який відповідає типу параметра.

Тип **string** мови Pascal – це фактично короткі рядки Delphi. Зарезервоване слово **string** у Delphi може забезпечити створення як Unicode-рядка, так і короткого рядка. Якщо при описі рядкової змінної після ідентифікатора **string** у квадратних дужках зазначена цілочислова константа, то в цьому випадку створюється короткий рядок. У противному разі тип створюваного рядка визначається налаштуванням компілятора. За умовчанням у Delphi 2009 тип **string** еквівалентний типу `UnicodeString`, що відповідає установці директиви компілятору `{$H+}`. Якщо необхідне трактування типу **string** як `ShortString`, застосовується директива `{$H-}`. Це ж можна здійснити в ICP Delphi. Для цього треба виконати команду Project ► Options (Проект ► Опції) і зробити налаштування у вікні, що відкриється за цією командою (у лівій частині вікна вибрати Delphi Compiler ► Compiling, після чого у правій частині вибрати пункт Syntax Options і у пункті Long strings by default встановити одне зі значень False або True.

При використанні типу `ShortString` для зберігання рядкового значення виділяється обсяг пам'яті в 255 байт, у кожному з байтів якого може зберігатися тільки один символ (тобто кожен елемент рядка фактично має тип `Char`). На самому початку області виділяється додатковий байт для зберігання символу, код якого дорівнює фактичній поточній довжині рядкового значення. Таким чином, під змінну типу `ShortString` насправді виділяється 256 байт. Якщо рядок інтерпретувати як масив, то у випадку короткого рядка елемент із індексом 0 – це символ, що визначає довжину рядка.

Аналогічно відводиться пам'ять та інтерпретується збережене значення при використанні типу **string**`[N]` – при цьому виділяється пам'ять обсягом `N+1` байт.

Максимальна довжина короткого рядка не може перевищити 255 або значення, зазначеного при його опису. Обсяг пам’яті, що відводиться під інші рядки не може перевищити 2 Гбайт.

Оголошення типізованої константи для будь-якого рядкового типу виконується у звичайний спосіб. Наприклад:

```
const
  s: string = 'Delphi 2009';
```

1.2. Присвоювання та введення/виведення рядків

Особливістю роботи з `UnicodeString`- та `AnsiString`-рядками є те, що змінні даного типу є вказівниками і пам’ять під них виділяється динамічно в міру необхідності, причому вони завершуються символом `#0`. Якщо рядок займав у пам’яті певний обсяг, то при його змінюванні пам’ять виділяється заново, і в неї переписується нове рядкове значення. При цьому пам’ять, у якій розташовувалося старе рядкове значення, звільняється не обов’язково – все залежить від того, чи не посилається на цю ділянку пам’яті інша рядкова змінна.

У випадку коротких рядків механізм рядкових присвоювань має деякі особливості:

1. Якщо довжини одержувача і джерела однакові, проблем немає:

```
Str6 := 'Delphi';
```

2. Якщо одержувач довший за джерело (`StrShort:='Delphi 2009'`), то фактична довжина одержувача встановлюється такою, як довжина рядка-джерела (таким чином, фактична довжина рядка `StrShort` дорівнюватиме 11, а її значенням буде `'Delphi 2009'`).

3. Якщо одержувач коротше джерела (`Str6 := 'Delphi 2009'`), то поточна довжина рядка-одержувача дорівнюватиме максимальній в оголошенні, а значенням буде усічене праворуч на потрібне число символів значення рядка-джерела (у цьому випадку `'Delphi'`).

У лівій та правій частинах оператора присвоювання може зустрітися типів `UnicodeString`, `AnsiString` та `ShortString`, причому тип лівої і правої частин не обов’язково повинні збігатись – має місце еквівалентність за присвоюванням для всіх цих типів. Присовювання довгих та широких рядків виконується з урахуванням механізму відведення пам’яті під них.

Виводити рядки можна присвоюванням властивості Caption видимих компонентів (наприклад, мітки), присвоюванням властивості Text однорядкового

редактора, вставкою у вікно багаторядкового редактора тощо. Уведення рядків звичайно здійснюють зчитуванням вмісту властивості Caption однорядкового редактора або вмісту рядка багаторядкового редактора.

1.3. Порівняння рядків та їхня конкатенація

За правилами Delphi з допомогою операцій =, <, <=, >, >=, <> дозволяється порівнювати рядкові змінні, константи і вирази з будь-якими максимальними та поточними довжинами. При цьому для встановлення факту рівності необхідно, щоб порівнювані об'єкти мали однакові фактичні довжини та точно збіжні значення символів. Коли в програмі порівнюються два рядки, насправді відбувається серія попарних порівнянь їхніх символів зліва направо до першої розбіжності або вичерпання одного з рядків. Меншим буде той рядок, у якого менший код першого незбіжного символу (поза залежністю від максимальних і поточних довжин порівнюваних рядків); якщо один рядок збігається з початком іншого, то більшим буде довший рядок (наприклад, є істинними співвідношення `'Іван' < 'Іваненко'` й `'довше' > 'довжина'`). Порівнювати можна дані будь яких рядкових типівж

Два рядки можуть бути зчеплені один з одним за допомогою операції конкатенації (зчеплення), позначуваної символом «+», причому у одному виразі можуть зустріьтися дані різних рядкових типів.

Так, виконання операторів

```
StrAnsi := 'Delphi';
Str := StrAnsi + ' 2009';
```

дасть результат `'Delphi 2009'`, що буде записаний у `Str`.

Операція конкатенації має більш вищий порівняно з операціями відношення =, <, <=, >, >=, <>.

1.4. Робота з окремими елементами рядка

При роботі з рядками можна звертатися до окремих їхніх елементів (символів) аналогічно тому, як це робиться при обробці елементів масиву. Окремі елементи рядкової змінної мають усі властивості змінної типу `Char`.

Можна коректувати будь-який елемент рядкової змінної, для чого у відповідному операторі досить зазначити ім'я рядкової змінної, слідом за яким у квадратних дужках задається номер її елемента (наприклад, `st[10] := 'f'`).

Елементи коротких рядків і UnicodeString- та AnsiString-рядків нумеруються від 1, а елементи інших рядкових типів – від 0).

У кожній короткій рядковій змінній є елемент із номером 0, у якому у вигляді символу зберігається поточна довжина рядка. Щоб дізнатися про фактичну довжину короткого рядка, потрібно застосувати функцію Ord до нульового елемента рядка (наприклад, k := Ord(StrShort[0]), де k – змінна цілого типу) або звернутися до функції Length. Нульовий елемент коротких рядків можна корегувати; при цьому буде змінюватися поточна довжина рядка (наприклад, оператор StrShort[0] := #65 задає поточну довжину рівною 65).

1.5. Процедури і функції для обробки рядків

Відразу зазначимо, що при обробці символьних масивів з нульовою базою і даних типу PChar перелічені нижче функції та процедури розглядають такого роду рядки як рядки з нумерацією елементів від 1 без порушення самої нумерації.

Замість операції конкатенації в рядкових виразах можливе використання функції Concat, причому як її параметр може виступати будь-яка кількість рядків:

```
ім'я_рядка := Concat(рядок1, рядок2, рядок3,...);
```

Значення, що вона повертає, – це рядок, який утвориться приєднанням значень параметрів: другого до кінця першого, третього до результату попереднього приєднання і т. д. Наприклад:

```
st := Concat(st1, st2, st3);
```

Для визначення поточної довжини рядка найчастіше застосовується функція Length, що повертає кількість символів, які містяться в даний момент у рядку, зазначеному як параметр:

```
змінна_цілого_типу := Length(рядок);
```

Для пошуку деякої послідовності символів у рядку уведено функцію Pos, що має такий формат:

```
p := Pos(шуканий_рядок, оброблюваний_рядок);
```

Обидва параметри можуть бути будь-якими рядковими виразам; значення, що повертає функція Pos, має тип Integer.

Якщо, наприклад, необхідно знайти рядок `st1` у рядку `st`, то при звертанні `p := Pos(st1, st)` у змінну цілого типу `p` буде записане число, що визначає ту позицію рядка `st`, у якій була виявлена перша поява рядка `st1`. Якщо `st1` відсутній у `st`, то результатом буде `0`.

Функція `Copy` використовується в Delphi для копіювання частини рядка. Можливий формат звертання до неї такий:

```
рядок1 := Copy(рядок, початкова_позиція, кількість_символів);
```

Функція повертає рядок, утворений зазначеним числом символів (параметр кількість_символів), розташованих у заданому рядковому виразі (параметр рядок) підряд, починаючи з позиції, номер якої визначається параметром початкова_позиція. Наприклад, оператор

```
st := Copy('останній', 2, 3);                    //st='стан'
```

записує у рядкову змінну `st` значення `'стан'`.

Якщо значення другого параметра задає позицію з номером більшим поточної довжини вихідного рядка, то повертається порожній рядок. При значенні другого параметра, меншому за 1, позиція початку копіювання визначається третім параметром. Якщо третій параметр дає вихід за поточну довжину рядка, то копіюються тільки реально існуючі символи (при від'ємному третьому параметрі результат – порожній рядок):

```
st := Copy('TForm', 2, Length('TForm'));  //st='Form'
```

Для видалення частини рядка використовується процедура `Delete`, що має три параметри:

```
Delete(рядок,поч_позиція,кількість_символів_що_видаляється);
```

Перший параметр – це рядкова змінна, два інших – будь-які цілочислові вирази.

Наприклад, наступні два оператори записують у `st` значення `'метр'`:

```
st := 'метеор';
Delete(st, 4, 2);
```

При застосуванні процедури `Delete` видаляється задане число символів, починаючи з зазначеної позиції, зі зсувом на позиції, що звільнилися, символів, розташованих правіше від вилучених, і зміною поточної довжини рядка. Якщо другий параметр перевищує поточну довжину або менший за 1, а також при від'ємному значенні третього параметра видалення не відбувається.

Процедура Insert виконує вставку рядка в деякий інший рядок, починаючи з зазначеної позиції, попередньо відсунувши вправо все, що заважає вставці. Формат процедури:

```
Insert(рядок_що_вставляється, рядок_що_приймає, поз_вставки);
```

У випадку коротких рядків, якщо максимальна довжина рядка, у який виконується вставка, менша результуючої довжини, то відбувається усікання результату праворуч. Якщо позиція вставки більша від поточної довжини рядка, то перший параметр приєднується до кінця другого. Значення позиції вставки, що менше за 1, приводить до вставки перед першим символом.

Наприклад, оператори

```
st := 'програвати';
Insert('мув', st, 7);
```

забезпечують запис в st значення 'програмувати'.

У деяких задачах виникає необхідність у перетворенні числових даних, записаних у рядковому виді, до якого-небудь числового формату. Для цих цілей у Delphi уведено процедуру Val, формат звертання до якої такий:

```
Val(рядок, числова_змінна, цілочислова_змінна);
```

Ця процедура присвоює другому аргументу числове значення, що зберігається у виразі, записаному як перший аргумент. Якщо під час перетворення не виявлена помилка, то в третій параметр записується 0; якщо помилка виявлена, то в третій параметр записується номер позиції першого помилкового символу рядка, а значення другого параметра буде дорівнювати нулю (у нультермінальних рядках при помилці повертається збільшений на 1 номер позиції першого помилкового символу). Другий параметр може мати будь-який числовий тип, і саме до цього типу буде виконуватися перетворення. У перетвореному рядку можуть бути лідируючі пробіли; пробіли ж наприкінці рядка зумовлюють діагностику помилки за допомогою третього параметра. Перетворення до цілого числа здійснюється завжди до першого помилкового символу з записом результату перетворення в другий параметр. Якщо ж перетворюється дійсне число, і виявляється помилка, то проводиться перетворення до цілого з ігноруванням десяткової точки та знака.

Наприклад, нехай у програмі є такі описи:

```
var
  r: Real;
  m, p: Integer;
  n: LongInt;
```

Нижче наведені приклади використання процедури Val:

```
Val('-12', m, p);              // m = -12; p = 0
Val('100023', n, p);           // m = 100023; p = 0
Val('-12e+004', r, p);         // r = -1.2e-05; p = 0
Val('-12e+004', n, p);         // n = 0; p = 4
Val('-1.2e+004 ', r, p);       // r = 12; p = 10
Val('   5623', n, p);          // n = 5623;  p = 0
Val('5623   ', n, p);          // n = 5623; p = 5
```

Зворотною стосовно процедури Val є процедура Str, формат якої

Str(арифметичний_вираз, рядок);

Процедура перетворює значення свого першого параметра в рядкову змінну, задану як другий параметр. Знак «+» у рядок не заноситься. Після арифметичного виразу може бути зазначений формат у вигляді одного або двох цілих додатних чисел, відокремлюваних одне від одного і від першого параметра двокрапкою. Перший параметр формату задає кількість знакомісць, що відводиться у рядку під запис числа (при цьому ліворуч рядок доповнюється пробілами), а другий – кількість символів після десяткової точки (можна задавати тільки при перетворенні дійсних значень). Елементи формату ігноруються, якщо їх значення не дозволяють здійснити перетворення.

Наприклад:

```
var
  st: string;
  ...
  st := Str(15 - 2 * 3, st);//st='9', поточна довжина 1
```

При роботі з символами в рядках, а також з окремими символами досить часто виникає необхідність у переході від малих латинських літер до великих. Для цього передбачено функцію UpCase, що має формат

символьна_змінна := UpCase(символ).

Функція не здійснює зворотного перетворення, так само як не перетворює жодних інші символи, у тому числі українські (російські) літери.

Описані в даному пункті процедури і функції перейшли в Delphi з мови Turbo Pascal. Однак у Delphi існує достатня кількість своїх процедур і функцій, орієнтованих на можливості і вимоги Windows та Linux. Насамперед, до цих підпрограм належать функції перетворення числових даних до рядкових і зворотно:

- `StrToCurr(`рядок`)` – повертає значення типу `Currency`;
- `StrToFloat(`рядок`)` – повертає значення типу `Extended` (враховується роздільник цілої та дробової частин, установлений за умовчанням або за допомогою системної змінної `DecimalSeparator`);
- `StrToInt(`рядок`)` – повертає значення типу `Integer`;
- `StrToInt64(`рядок`)` – повертає значення типу `Int64`;
- `FloatToStr(`числовий_вираз`)` – перетворить дійсне число до рядкового подання, повертаючи значення типу **`string`** з розділь-ником цілої та дробової частин, що задаються за умовчанням або за допомогою системної змінної `DecimalSeparator`;
- `IntToStr(`цілочисловий_вираз`)` – перетворює задане першим параметром значення в рядок, повертаючи значення типу **`string`**;
- `Int64ToStr(`цілочисловий_вираз`)` – працює аналогічно функції `IntToStr`.

Зазначимо, що при звертанні до функцій `StrToCurr`, `StrToFloat`, `StrToInt` і `StrToInt64` у рядку не повинно бути початкових і кінцевих узагальнених пробільних символів, для видалення яких можна використовувати функцію

`Trim(`рядок`)`,

що повертає значення типу **`string`**. Початкові та кінцеві пробіли в параметрі функцій `StrToCurr` і `StrToFloat` та початкові пробіли в параметрі функцій `StrToInt` і `StrToInt64` можуть бути присутніми.

До деяких із зазначених вище функцій ми вже зверталися в наведених раніше прикладах розв'язання задач.

Функції

`StrToIntDef(`рядок, цілочисловий_вираз`)`

та

`StrToInt64Def(`рядок, цілочисловий_вираз`)`

здійснюють перетворення рядка з видачею у випадку помилкового перетворення значення, що задається другим параметром.

Визначена також функція `FloatToStrF`, що має такий формат звернення:

`FloatToStrF(`числовий_вираз, формат, точність, цифри`)`.

Ця функція перетворює дійсне число в рядок відповідно до формату, що задається другим параметром. Другий параметр може набувати одного зі значень наступного переліченого типу, що визначений у модулі `SysUtils`:

```
type
  TFloatFormat = (ffGeneral, ffExponent, ffFixed,
                  ffNumber, ffCurrency).
```

Зміст третього та четвертого параметрів залежить від значення другого параметра. Так, якщо як другий параметр задати значення `ffFixed`, то третій параметр визначає кількість цифр у запису числа, а четвертий – кількість цифр у дробовій частині; якщо другий параметр дорівнює `ffExponent`, то третій параметр задає кількість цифр у запису мантиси, а четвертий – кількість цифр у запису десяткового порядку. Формат `ffGeneral` переключає перетворення на один з форматів `ffFixed` і `ffExponent` залежно від значення числа. Формат `ffCurrency` є грошовим форматом (ураховує знак національної грошової одиниці), а формат `ffNumber` вживається при виведенні великих чисел і забезпечує поділ тисяч.

Визначено й інші функції взаємного перетворення числових і рядкових даних, а також перетворення рядкових даних до даних типу `TDataTime` і зворотно.

У мові визначений також ряд функцій перетворення регістра літер, що входять у рядок (повертають значення типу **`string`**):

- `AnsiLowerCase(`рядок`)` – повертає перетворений рядок, у якому всі літери (у тому числі й національного алфавіту) замінені на малі;
- `AnsiUpperCase(`рядок`)` – повертає перетворений рядок, у якому всі літери (у тому числі й національного алфавіту) замінені на великі;
- `LowerCase(`рядок`)` – повертає перетворений рядок, у якому всі латинські літери замінені на малі (інші символи не перетворюються);
- `UpperCase(`рядок`)` – повертає перетворений рядок, у якому всі латинські літери замінені на великі (інші символи не перетворюються).

Досить зручною є функція

`StringOfChar(`символ, цілочисловий_вираз`)`,

яка формує рядок із символів, що задаються першим параметром, у кількості, що визначається другим параметром.

Для порівняння рядків у Delphi визначено функції, що повертають значення типу `Integer` і мають параметри типу **`string`**:

- `AnsiCompareStr(`рядок1, рядок2`)` – порівняння рядків у визначеній регіональним стандартом системі сортування символів з урахуванням регістру (у тому числі для символів національного алфавіту);
- `AnsiCompareText(`рядок1, рядок2`)` – порівняння рядків у визначеній регіональним стандартом системі сортування символів без урахування регістру (у тому числі для символів національного алфавіту);
- `CompareStr(`рядок1, рядок2`)` – порівняння рядків за кодами символів з урахуванням регістру;
- `CompareText(`рядок1, рядок2`)` – порівняння рядків за кодами символів без урахування регістру латинських літер (регістр літер національного алфавіту враховується).

Значення, що повертає кожна з цих функцій, визначається таким правилом:

- результат порівняння – будь-яке значення менше `0`, якщо рядок1 < рядок2;
- результат порівняння дорівнює `0`, якщо рядок1 = рядок2;
- результат порівняння – будь-яке значення більше `0`, якщо рядок1 > рядок2.

Функції `AnsiCompareStr` та `AnsiCompareText` орієнтовані на впорядкування символів, яке визначається не таблицею ANSI-кодів, а регіональними стандартами, відповідно до яких великі та малі літери національних алфавітів розташовуються поруч, причому велика літера передує малій (це характерно для більшості національних алфавітів).

Наступні функції перевіряють два рядки на рівність:

`AnsiSameStr(`рядок1, рядок2`)`,
`AnsiSameText(`рядок1, рядок2`)`,
`SameText(`рядок1, рядок2`)`.

Ці функції повертають булівське значення `True`, якщо параметри виклику рівні, і значення `False` у противному разі. При цьому друга функція при порівнянні не враховує регістр літер як латинського, так і національного алфавіту. Функції `AnsiSameStr` та `AnsiSameText` орієнтовані на впорядкування символів, визначене регіональним стандартом, у той час як функція `SameText` орієнтована на впорядкування символів за їх кодами.

2. ПРИКЛАДИ ПРОГРАМ

Для розв'язання задач, що сформульвані нижче, створимо форму, розмістивши на ній однорядковий редактор з міткою над ним, багаторядковий редактор та звичайну кнопку і здійснивши такі зміни значень деяких властивостей цих компонентів:

Мітка:

Name — lbOutput1

Caption — Уведіть рядок

Однорядковий редактор:

Name — edInput1

Text — очистимо

Багаторядковий редактор:

Name — mmOutput1

Lines — очистимо

Кнопка:

Caption — Увести

Приклад 1. Дано рядок *s*, символ *c*. Встановити, скільки разів символ *c* зустрічається в рядку *s*.

Розв'язання. Оголосимо в секції **public** опису класу `TForm1` два поля:

```
s: string;
c: Char;
```

Задачу розв'язує такий оброблювач події OnClick кнопки Button1:

```
procedure TForm1.Button1Click(Sender: TObject);
var
  i, Count_c: Integer;
begin
  edInput1.SetFocus;
  if Tag = 0 then begin
    s := edInput1.Text;
    lbOutput1.Caption := 'Уведіть символ';
    Button1.Caption := 'Run';
  end
  else begin
    c := edInput1.Text[1];
    Count_c := 0;
    for i := 1 to Length(s) do  // Цикл за елементами рядка
      if s[i] = c then Inc(Count_c);
```

```
      mmOutput1.Lines.Add('Count = ' + IntToStr(Count_c));
      Button1.Visible := False;
      lbOutput1.Visible := False;
      edInput1.Visible := False;
  end;
  Tag := Tag + 1;
end;
```

Приклад 2. Дано рядок *s* і символ *c*. Видалити з рядка *s* усі входження символу *c*.

Розв'язання. Оголосимо в секції **public** опису класу TForm1 два поля:

```
s: string;
c: Char;
```

Задачу розв'язує такий оброблювач події OnClick кнопки Button1:

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  edInput1.SetFocus;
  if Tag = 0 then begin
    s := edInput1.Text;
    lbOutput1.Caption := 'Уведіть символ';
    Button1.Caption := 'Run';
  end
  else begin
    c := edInput1.Text[1];
    mmOutput1.Lines.Add('Оброблюваний рядок');
    mmOutput1.Lines.Add(s);
    while Pos(c, s) <> 0 do   // Поки у s є шуканий символ,
      Delete(s, Pos(c, s),1);       // здійснюємо видалення
      mmOutput1.Lines.Add('Результуючий рядок');
      mmOutput1.Lines.Add(s);
      Button1.Visible := False;
      lbOutput1.Visible := False;
      edInput1.Visible := False;
  end;
  Tag := Tag + 1;
end;
```

Приклад 3. Дано рядок *s*, символ *c*. Видалити з рядка *s* усі символи, починаючи з першого входження і закінчуючи останнім входженням символу *c*.

Приклад 3. Дано рядок *s* і символ *c*. Видалити з рядка *s* усі входження символу *c*.

Розв'язання. Оголосимо в секції **public** опису класу TForm1 два поля:

```
s: string;
c: Char;
```

Задачу розв'язує такий оброблювач події OnClick кнопки Button1:

```
procedure TForm1.Button1Click(Sender: TObject);
var
  pL, pR: Integer;    // Позиції лівого і правого входження
                                                  // символу
begin
  edInput1.SetFocus;
  if Tag = 0 then begin
    s := edInput1.Text;
    lbOutput1.Caption := 'Уведіть символ';
    Button1.Caption := 'Run'
  end
  else begin
    c := edInput1.Text[1];
    mmOutput1.Lines.Add('Оброблюваний рядок');
    mmOutput1.Lines.Add(s);
    pL := Pos(c, s);   // Позиція першого входження символу
    if pL <> 0 then begin  // Якщо символ присутній у рядку
      pR := Length(s);
      while pR >= pL do  // Цикл пошуку останнього входження
      begin
        if s[pR] = c then Break;
        Dec(pR);
      end;
      Delete(s, pL, pR - pL +1); // Видалення частини рядка
    end;
    mmOutput1.Lines.Add('Результуючий рядок');
    mmOutput1.Lines.Add(s);
    Button1.Visible := False;
    lbOutput1.Visible := False;
    edInput1.Visible := False;
  end;
  Tag := Tag + 1;
end;
```

3. ЗАВДАННЯ НА ЛАБОРАТОРНУ РОБОТУ

За час, відведений для виконання лабораторної роботи (2 академічні години), студент повинен:

1. Розробити алгоритм розв'язання задачі, запропонованої для програмування.
2. Здійснити проектування форми для функціонування розроблюваної програми.
3. Здійснити програмну реалізацію розробленого алгоритму.
4. Здійснити відлагодження програми, виправивши синтаксичні та логічні помилки.
5. Підібрати тестові дані для перевірки програми, включаючи виняткові випадки.
6. Оформити звіт до лабораторної роботи.
7. Відповісти на контрольні запитання.
8. Здати викладачу працездатну програму з демонстрацією її роботи на декількох варіантах вихідних даних.

4. ВАРІАНТИ ЗАДАЧ

1. Дано рядок. Чи правда, що він складається тільки з латинських літер?
2. Дано рядки символів s і g. Скільки разів рядок g зустрічається в рядку s?
3. Дано рядок. Визначити кількість слів, що містяться в ньому. У загальному випадку під терміном «слово» будемо розуміти будь-яку послідовність символів, що не містить узагальнених пробільних символів, тобто символів з кодами, меншими за 33.
4. Дано рядок. Визначити довжину найкоротшого зі слів, що містяться в ньому.
5. Дано рядок. Вивести всі його слова без повторення.
6. Дано рядки s, g і натуральне число k ($k \le L+1$, де L – кількість символів у рядку s). Виконати вставку рядка g в k-ту позицію рядка s. Якщо $k = L+1$, рядок g повинен бути дописаний до рядка s праворуч.
7. Дано рядок. Чи правда, що він однаково читається зліва направо і справа наліво, тобто є *паліндромом*?
8. Дано рядок, що складається з латинських літер (великих і малих) та пробілів. Чи правда, що він (без урахування пробілів) однаково читається зліва направо і справа наліво?

9. Дано рядок. Визначити кількість слів, що містяться в ньому і є паліндромами.
10. Дано рядок s, що містить n символів $s_1, s_2, \ldots, s_n$. Якщо цей рядок є паліндромом, тобто $s_1 = s_n$, $s_2 = s_{n-1}, \ldots$, то залишити його без зміни, інакше – доповнити цей рядок справа його «дзеркальним» відбиттям без повторення останнього символу (s_1, s_2, …, s_{n-1}, s_n, s_{n-1}, …, s_2, s_1).
11. Дано рядок, що містить одне або декілька слів, розділених одним чи кількома пробілами. Сформувати новий рядок, що містить розділені одним пробілом ці ж слова, але розташовані у зворотному порядку.
12. Дано рядки символів s і g. Приєднати рядок g праворуч до рядка s. Якщо початок рядка g збігається з кінцем рядка s, при приєднанні рядків виключити таке накладання символів, видаливши найбільшу їх кількість.

    Наприклад:

    s: Delphi is one of progr

    g: rogramming languages.

    Результуючий рядок: Delphi is one of programming languages.

    Тут видалений підрядок rogr, хоч можна було видалити тільки r.
13. Дано рядки символів s і g. Чи можна з символів, що містяться в рядку s, сформувати рядок g?

    а) Кожен із символів рядка може бути використаний необмежену кількість разів.

    б) Символ, розміщений у будь-якій з позицій рядка s, може бути використаний тільки один раз.
14. Замінити у рядку всі ланцюжки символів Pascal на сполучення символів C++.
15. Дано рядок, що містить десяткові цифри. Замінити кожну цифру її назвою англійською мовою, розділяючи кожне зі слів одним пробілом (0 – zero, 1 – one, 2 – two, 3 – three, 4 – four, 5 – five, 6 – six, 7 – seven, 8 – eight, 9 – nine).
16. Дано рядки s і g. Видалити з рядка s підрядок, що збігається з g. Виконати видалення:

    а) першого входження рядка g;

    б) останнього входження рядка g;

    в) усіх входжень рядка g (наприклад, при видаленні з рядка abzwzwzwbzwzcdzwzwz рядка zwz має бути отриманий рядок abwzwbcdwz).
17. Рядок складається зі слів (ланцюжків символів, відділених один від іншого одним або декількома пробілами). Вибрати ті зі слів, які є записом цілих чисел, а ті, що залишилися, розсортувати на слова, які являють собою запис

дійсних чисел, і слова, що не є записом чисел тією мовою програмування, якою пишеться програма. Сформувати відповідно три рядки, слова в яких розділені одним пробілом.

18. Дано натуральне число *n*. Одержати рядкове подання цього числа у вигляді послідовності цифр і пробілів, що відокремлюють групи по три цифри, починаючи праворуч.
19. Дано рядок, що складається зі слів, розділених одним або декількома пробілами. Видалити з кожного слова рядка всі попередні входження його останнього символу без видалення інших символів.

5. КОНТРОЛЬНІ ЗАПИТАННЯ

1. Які символьні типи визначені у Delphi?
2. Що таке рядкові дані?
3. Які рядкові типи існують у Delphi?
4. У чому особливості коротких рядків?
5. У чому особливості рядків типу `AnsiString`?
6. У чому особливості рядків типу `UnicodeString`?
7. Які рядки за умовчанням ставляться у відповідність типу **`string`**?
8. Як можна впізнати фактичну довжину рядка?
9. Як проводиться порівняння даних типу **`string`**?
10. Як здійснюється перетворення числових даних до рядкового подання?
11. Як можна перетворити рядок у число?
12. Як можна здійснювати уведення ти виведення рядкових даних?
13. Як здійснюється копіювання рядків?
14. Чи можна виконати копіювання частини рядка? При позитивній відповіді опишіть иетодику копіювання.
15. Чи можна здійснити видалення частини рядка? При позитивній відповіді опишіть иетодику видалення.
16. Чи можна здійснити вставку одного рядка у інший рядок? При позитивній відповіді опишіть иетодику вставки.

СПИСОК ЛІТЕРАТУРИ

1. Безменов, М. І. Основи програмування в середовищі Delphi : навч. посіб. / М. І. Безменов. – Х. : НТУ «ХПІ», 2010. – 608 с.
2. Кэнту, М. Delphi 7 : Для профессионалов / М. Кэнту – СПб. : Питер, 2004. – 1101 с.
3. Архангельский, А. Я. Программирование в Delphi 6 / А. Я. Архангельский. – М. : БИНОМ, 2002. – 1120 с.
4. Дарахвелидзе, П. Г. Программирование в Delphi 7 / П. Г. Дарахвелидзе, Е. П. Марков. – СПб. : БХВ-Петербург, 2003. – 784 с.
5. Культин, Н. Б. Основы программирования в Delphi 7 / Н. Б. Культин. – СПб.: БХВ-Петербург, 2003. – 608 с.
6. Пестриков, В. М. Delphi на примерах / В. М. Пестриков, А. Н. Маслобоев. – СПб. : БХВ-Петербург, 2005. – 496 с.
7. Ремкеев, А. А. Курс Delphi для начинающих. Полигон нестандартных задач / А. А. Ремкеев, С. В. Федотова. – М. : СОЛОН-Пресс, 2006. – 360 с.
8. Митчелл, К. Керман. Программирование и отладка в Delphi : учебный курс / Митчелл К. Керман. – М. : Вильямс, 2004. – 720 с.
9. Парижский, С. М. Delphi : Только практика / С. М. Парижский. – К. : МК-Пресс, 2005. – 208 с.
10. Культин, Н. Б. Основы программирования в Delphi 2007 / Н. Б. Культин. – СПб. : БХВ-Петербург, 2008. – 480 с.

Навчальне видання

Методичні вказівки
до лабораторної роботи
«Рядкові дані у Delphi»
з курсу «Програмування» для студентів напряму 6.040302 – Інформатика
(спеціалізація «Соціальна інформатика»)

Укладач БЕЗМЕНОВ Микола Іванович

Відповідальний за випуск О. С. Куценко
Роботу до видання рекомендував О. В. Горелий

За авторською редакцією

План 2011 р., поз. 8 / 74-11

Підп. до друку 23.05.2011 р. Формат 60 × 84 $^{1}/_{16}$. Папір офісний.
Riso-друк. Гарнітура Таймс. Ум. друк. арк. 1,1. Наклад 50 прим.
Зам. № 162. Ціна договірна.

Видавничий центр НТУ «ХПІ».
Свідоцтво про державну реєстрацію ДК № 3657 від 24.12.2009 р.
61002, Харків, вул. Фрунзе, 21

Друкарня НТУ «ХПІ», 61002, Харків, вул. Фрунзе, 21